**第四章 使用Kolla部署Pike版本的OpenStack-allinone云平台**

**本节所讲内容：**

**4.1 openstack 概述**

**4.2 openstack的作用**

**4.3 openstack版本命名规则**

**4.4 openstack各组件关系**

**4.5 Kolla概述和openstack所有结点linux系统初始配置**

**4.6 安装kolla-ansible**

**4.7 自定义kolla-ansible安装openstack的相关配置文件**

**4.8 开始基于kolla-ansible安装openstack私有云**

****

**4.1 openstack 概述**

**openstack概述 ：**

**OpenStack是一个NASA美国国家航空航天局和Rackspace合作研发的，以Apache许可证授权，并且是一个自由软件和开放源代码项目。**

**Open是开放，Stack则是堆砌之意，合起来就是将众多的功能服务堆积起来的集合，让人们通过Openstack云计算项目，能够将诸如计算能力、存储、网络和软件等资源抽象成服务，以便让用户可以通过互联网远程来享用，付费的形式也变得因需而定，调整方便，拥有极强的虚拟可扩展性，是公共和私有云的建设与管理软件中的优秀开源项目。**

**4.2 openstack的作用**

**openstack的作用：**

**OpenStack是一个云平台管理的项目，它不是一个软件。这个项目由几个主要的组件组合起来完成一些具体的工作。**

**OpenStack是一个旨在为公共及私有云的建设与管理提供软件的开源项目。它的社区拥有超过130家企业及1350位开发者，这些机构与个人都将OpenStack作为基础设施即服务（简称IaaS）资源的通用前端。**

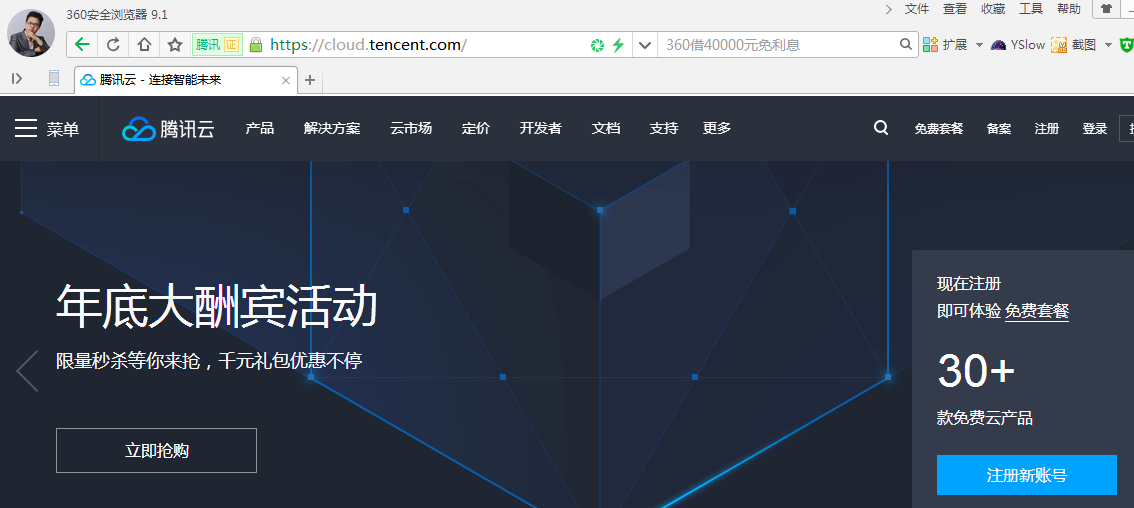
**1、IaaS（Infrastructure as a Service），即基础设施即服务。**

**先生，你用过这个服务吗？**

**消费者通过Internet 可以从完善的计算机基础设施获得服务。这类服务称为基础设施即服务。基于 Internet 的服务（如存储和数据库）是 IaaS的一部分。**

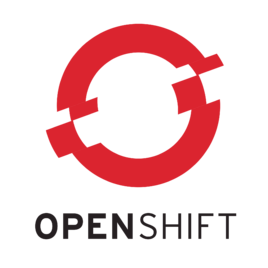
**比如： 在腾讯云上买一台云主机（8个CPU，32G，5T硬盘云主机等）**

**例： 腾讯云： https://cloud.tencent.com/**



**2、PaaS是Platform-as-a-Service的缩写，意思是开发**[**平台**](http://baike.baidu.com/view/58664.htm)**即**[**服务**](http://baike.baidu.com/subview/133203/11217422.htm)**。 把**[**服务器**](http://baike.baidu.com/view/899.htm)**平台作为一种服务提供的商业模式。**

**例：OpenShift是红帽的云开发平台即服务（PaaS）。 OpenShift的logo如下:**

****

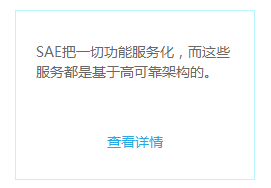
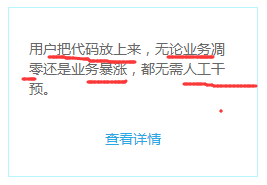
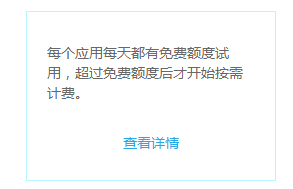
**OpenShift是自由和开放源码的云计算平台，它可以使开发人员能够创建、测试和运行他们的应用程序，并且可以把它们部署到云中。Openshift广泛支持多种编程语言和框架，如Java，Ruby和PHP等。另外它还提供了多种集成开发工具如Eclipse integration，JBoss Developer Studio和 Jenkins等。OpenShift 基于一个开源生态系统为移动应用，数据库服务等，提供支持。**

**哪些厂商在做PaaS平台？**

**例2：新浪的SAE PaaS云平台，国内：http://www.sinacloud.com/sae.html**

****

**PaaS 的 4大特点：**

****

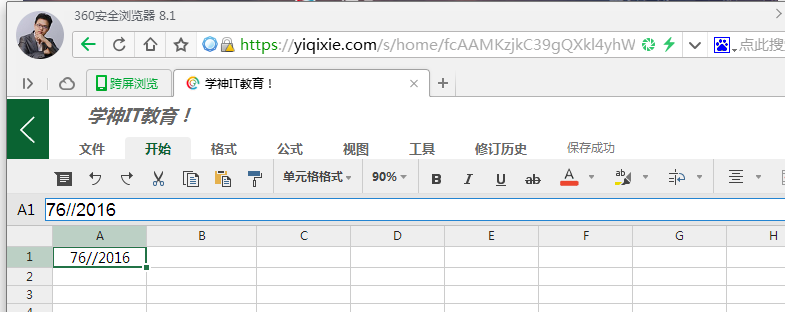
**注：Iaas干了硬件工程师的活 , paas干了运维的活**

**3、SaaS是Software-as-a-Service（**[**软件即服务**](http://baike.baidu.com/view/1559138.htm)**）的简称。**

**例： https://yiqixie.com 中国的一起写， 多人编辑一个word或execle**

**国外：docs.google.com或 微软的office 365**

****

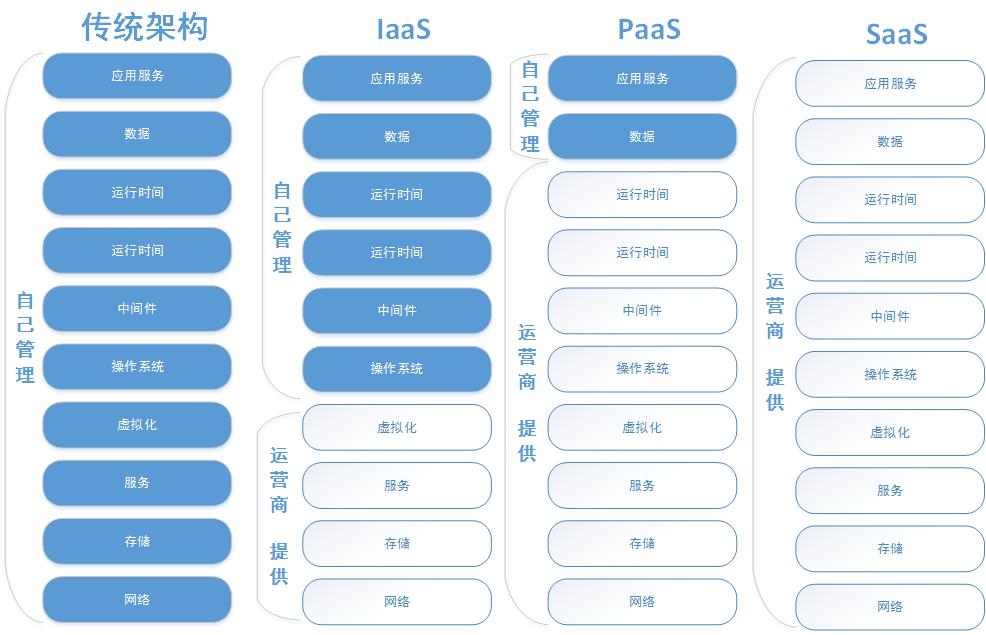
****

**注：有SaaS平台， 不需要运维，不需要开发。**

**要求：你越来越专业！**

**总结：**

**三种云平台架构： IAAS ,PAAS ,SAAS 都是不同的商业模式。**

****

**按拥有者分类：**

**公有云（Public Cloud）、私有云（Private Cloud）、混合云（Hybrid Cloud）**

**按照技术厂商分类：**

**VMware vSphere、微软云计算解决方案、亚马逊AWS、OpenStack等**

**注：国内云平台使用opensctack二次开发比较多。**

**4.3 openstack版本命名规则**

**4.3.1 Openstack命名方式**

**版本发布以A-Z字母顺序来发布，很好记。**

**例:**

**2010年10月 发布‘Austin’**

**2011年2月 发布‘Bexar’ [地名] [美国] 贝尔;**

**2011年4月 发布‘Cactus’ [地名] [美国] 卡克特斯;**

**2011年7月 发布‘Diablo’**

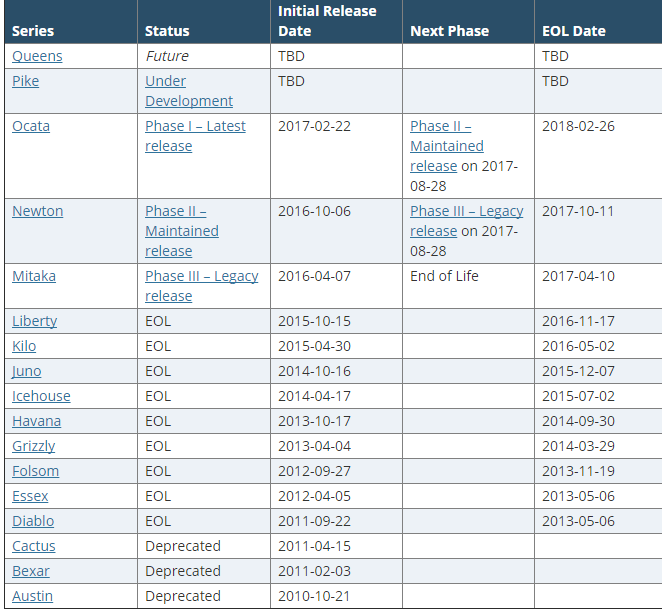
**2012年4月 发布‘Essex’**

**2012年10月 发布‘Folsom’**

**2013年4月 发布‘Grizzly’**

**2013年10月 发布Havana**

**2014年4月 发布‘Icehouse’**

****

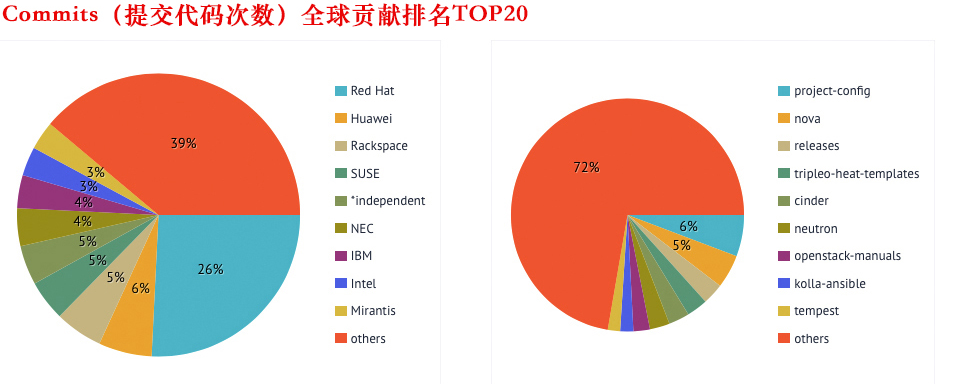
**https://releases.openstack.org/**

**互动：如果到了Z以后，怎么起名？**

**AA。。。 AB。。**

**磁盘起名到sdz后，怎么办？ 不懂： 1 懂：6 答： sda。。。。sdz -》 sdaa 。。sdab**

**查看最新版本openstack yum源：https://repos.fedorapeople.org/repos/openstack/**

**OpenStack Pike版本发布 看中国公司贡献率排行**

**中国公司在OpenStack社区的贡献大致可以看出如下格局：**

****

**4.4 openstack各组件关系**

**4.4.1 openstack核心组成主要有:**

**核心组件**

* + **Keystone（身份认证）**
  + **Nova（计算）**
  + **Neutron（网络）**
  + **Glance（镜像存储）**
  + **Cinder（块存储）**
  + **Swift（对象存储）**
  + **Horizon（web UI界面）**
  + **Ceilometer（计量）**
  + **Heat（部署编排）**
  + **Trove（数据库）**

**下面详细说明：**

**身份认证（Keystone）：统一的授权、认证管理。所有组件都依赖于Keystone提供3A(Account, Authentication, Authorization)服务。  
 例1： 3A认证**

**1、认证(Authentication)，验证用户的身份与可使用的网络服务；**

**2、授权(Authorization)：依据认证结果开放网络服务给用户；**

**3、计帐(Accounting)：记录用户对各种网络服务的用量，并提供给计费系统。整个系统在网络管理与安全问题中十分有效。**

**比如：宽带收费就是3A认证的典型例子：输入帐号密码（认证）-》开10M带宽（授权）-》在营业厅（计帐）**

**计算管理（Nova）：Nova是OpenStack云中的计算组织控制器。Nova自身并没有提供任何虚拟化能力，相反它使用libvirt API来与被支持的虚拟技术Hypervisors交互。如：kvm、Xen、VMware等虚拟化技术。**

**Neutron（网络）：实现虚拟机的网络资源管理如网络连接、ip管理、公网映射**

**镜像管理（Glance）：  主要存储和管理系统镜像。 cento镜像**

**块存储（Cinder）：为虚拟机提供存储空间。 比如硬盘，分区，目前支持ip-san、fc-san等。  
对象存储（Swift）：OpenStack Swift 开源项目提供了弹性可伸缩、高可用的分布式对象存储服务，适合存储大规模非结构化数据。通过key/value的方式实现对文件的存储，现在的云盘就是这样的，和MFS，HDFS类似**

**注：如果客户需要一个1000T的存储空间，使用Cinder或Glance就不行，效率太低。这时就用Swift。**

**界面（Horizon）：安装好后，openstack的web界面控制台DashBoard**

**Ceilometer（计量）：Ceilometer是OpenStack中的一个子项目，它像一个漏斗一样，能把OpenStack内部发生的几乎所有的事件都收集起来，然后为计费和监控以及其它服务提供数据支撑。**

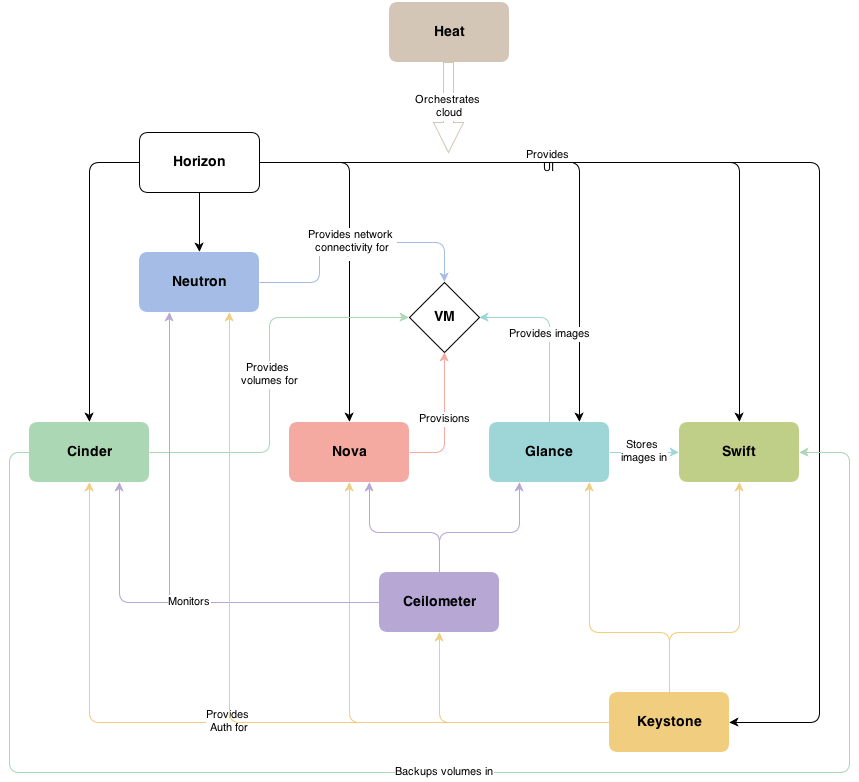
**Ceilometer [sɪ'lɒmɪtə] 云幂测量仪**

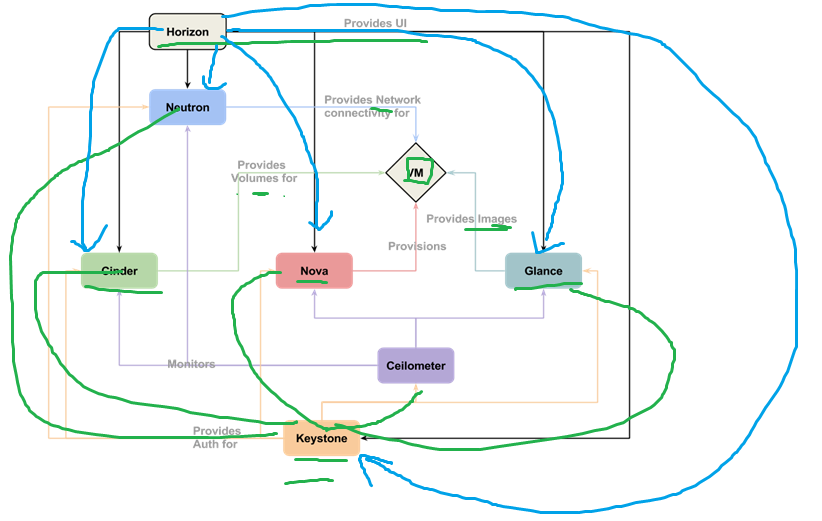
**Heat（部署编排）：是一个编排引擎，它可以基于文本文件形式的模板启动多个复合云应用程序（这些文件可以被视为代码）。简单来说，Heat为OpenStack用户提供了一种自动创建云组件（如网络、实例、存储设备等）的方法。**

**[hɪt]**

**Trove（数据库）：为关系型数据库和非关系型数据库引擎提供可扩展的和可靠的云数据库服务，并继续改进其功能齐全、可扩展的开源框架。**

**Trove [trəʊv] 宝库**

****

****

**Openstack的网络模式有5种**

**Local模式：一般测试时使用，只需一台物理机即可。**

**GRE模式：隧道模式， VLAN数量没有限制，性能有点问题。**

**Vlan模式：vlan数量有4096的限制**

**VXlan模式：vlan数量没有限制，性能比GRE好。**

**Flat模式：管理员创建租户直接到外网，不需要NAT。**

**扩展：VXLAN概述**

**VXLAN是由思科与VMware提出的。**

****

**LAN 局域网**

**VLAN 虚拟局域网**

**VXLAN 虚拟扩展局域网**

**VXLAN (Virtual Extensible LAN，虚拟扩展局域网) 它是一种在UDP中封装MAC的简单机制，可以创建跨多个物理IP子网的虚拟2层子网**

**比如： 我在北京，你在上海，通过vxlan技术，可以让我们在同一个局域网中。 （有点vpn隧道）**

**为什么我们需要这种新技术?**

**现在不只一种标准可以在IP中封装MAC(包括EtherIP和GRE通道桥接)，但是它们都无法做到VLAN逻辑标记，因此您只能使用最多4096个不同的VLAN。即使可以使用这些标准来分隔逻辑网，您也必须(在负载中)深入分析MAC头信息，以查找出虚拟分片ID。而VXLAN通过一个24位分片ID，在一个数据中心内它能部署百万个虚拟网。**

**而且，VXLAN数据包格式很容易在硬件中实现，所以它以后也许可以更紧密地与物理网络设备集成。**

**何时需要VXLAN?**

**在需要大量逻辑网时，才应该考虑使用VXLAN。如果只有少量逻辑网，那么就还是应该使用经过时间考验的技术，如VLAN。 你的逻辑网不超过4096个，都用VLAN。**

**VXLAN硬件产品：H3C S6800数据中心以太网汇聚交换机**

****

**H3C S6800系列交换机支持VXLAN (Virtual Extensible LAN，虚拟扩展局域网)， VXLAN通过将虚拟机发出的数据包封装在UDP中，并使用**[**物理网络**](http://zhidao.baidu.com/ues/index?word=%E7%89%A9%E7%90%86%E7%BD%91%E7%BB%9C&fr=qb_search_exp&qb_ues_exp=1&ie=utf8)**的IP/MAC作为outer-header封装后在物理IP网上传输，到达目的地后由隧道终结点解封并将数据发送给目标虚拟机，解决了地理分散的**[**数据中心**](http://www.cioage.com/col/1335/)**之间远距离虚拟机迁移问题。**

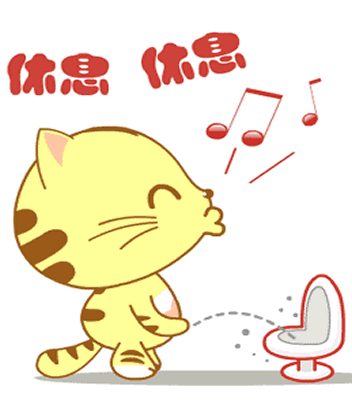
**VLAN与VXLAN之间有何区别：VXLAN显然更具可扩展性(4,096个VLAN网 vs 1600万个VXLAN网)**

**openstack项目相关资源获取**

* + **http://www.openstack.org**
  + **https://github.com/openstack**
  + **http://docs.openstack.org/**
  + **http://www.openstack.cn openstack中文**

**4.4.2 OpenStack部署方法，主要有以下5种：**

* **社区手册**
  + **http://docs.openstack.org**
* **RDO**
  + **https://www.rdoproject.org（http://openstack.redhat.com）**
* **RedHat Enterprise Linux OpenStack Platform （E210 考试）**
  + **http://www.redhat.com/en/technologies/linux-platforms/openstack-platform**
* **Mirantis（Fuel）**
  + **https://www.mirantis.com**
* **高级定制**
  + **Puppet、Chef**
* **kolla 基于docker安装openstack ，把openstack每个组件做成docker实例**



**4.5 Kolla概述和openstack所有结点linux系统初始配置**

**kolla是openstack下面用于自动化部署的一个项目，它基于docker和ansible来实现，docker主要负责镜像制作，容器管理。而ansible主要负责环境的部署和管理。**

**Kolla实际上是分为两大块的，一部分，Kolla提供了生产环境级别的镜像，涵盖了Openstack用到的各个服务，另一部分是自动化的部署，也就是上面说的ansible部分。最开始两个部分是在一个项目中的（也就是Kolla），从O版本开始将两个部分独立开来，Kolla项目用来构建所有服务的镜像，Kolla-ansible用来执行自动化部署。**

**4.5.1 linux系统环境配置**

**1、关闭Selinux和防火墙**

**[root@xuegod63 ~]# vim /etc/selinux/config**

**SELINUX=disabled**

**[root@xuegod63 ~]# reboot #如果原来的系统开着selinux，那么需要重启，才能关闭selinux**

**2、关闭Firewalld**

**[root@xuegod63 ~]# systemctl stop firewalld**

**[root@xuegod63 ~]# systemctl disable firewalld**

**[root@xuegod63 ~]# systemctl status firewalld**

**3、安装 Epel源**

**[root@xuegod63 ~]# mv /opt/CentOS-\* /etc/yum.repos.d/ #这个在大家的机器上不需要执行。 因为我的centos自带的yum源配置文件，被移走了**

**[root@xuegod63 ~]# yum install epel-release -y**

**4、配置 Hostname**

**[root@xuegod63 ~]# cat /etc/hostname**

**xuegod63.cn**

**5、配置/etc/hosts**

**[root@xuegod63 ~]# cat /etc/hosts**

**127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4**

**::1 localhost localhost.localdomain localhost6 localhost6.localdomain6**

**192.168.1.63 xuegod63.cn xuegod63**

**6、同步时间**

**[root@xuegod63 ~]# yum install ntp**

**[root@xuegod63 ~]# systemctl enable ntpd.service**

**[root@xuegod63 ~]# systemctl start ntpd.service**

**7、配置 pip 镜像源，方便快速下载python库（这一步很重要）**

**[root@xuegod63 ~]# mkdir ~/.pip**

**[root@xuegod63 ~]# vim ~/.pip/pip.conf #写入下以内容**

**[global]**

**index-url = http://mirrors.aliyun.com/pypi/simple/**

**[install]**

**trusted-host=mirrors.aliyun.com  
4.5.2 配置网卡信息**

**在xuegod63上添加一个新的网卡，类型是bridge模式**

| **IP地址** | **网络类型** | **网卡** | **在openstack网络中的作用** |
| --- | --- | --- | --- |
| **192.168.1.63** | **bridge** | **ens33** | **openstack内部管理网络 (managment nework) 日后Horizon web界面访问，就是通过这个IP地址** |
| **无IP（不能配置IP）** | **bridge** | **ens38** | **外部网络 (external network) ，让neutron的br-ex 绑定使用，openstack中的虚拟机是通过这块网卡和外网通信的** |

**注：如果在服务器直接安装，两块网卡的IP，就算是同一个网段或者不同网段都可以的。**

**生成ens38的网卡配置文件**

**[root@xuegod63 ~]#cd /etc/sysconfig/network-scripts/**

**[root@xuegod63 network-scripts]# cp ifcfg-ens33 ifcfg-ens38**

**[root@xuegod63 network-scripts]# vim ifcfg-ens38**

**TYPE="Ethernet"**

**BOOTPROTO="none"**

**NAME="ens38"**

**DEVICE="ens38"**

**ONBOOT="yes"**

**检查一下ens33网卡配置文件，发现在新添加网卡，重启系统后，启动协议变了，现在改回来。**

**[root@xuegod63 network-scripts]# vim ifcfg-ens33**

**改：BOOTPROTO="dhcp"**

**为：BOOTPROTO="none"**

**[root@xuegod63 network-scripts]# service network restart**

**4.5.3 安装基础包和docker服务**

**1、安装基础包**

**[root@xuegod63 ~]# yum install python-devel libffi-devel gcc openssl-devel git python-pip -y**

**[root@xuegod63 ~]# pip install -U pip #升级一下pip，不然后，后期安装会报警告**

**[root@xuegod63 ~]# yum install -y yum-utils device-mapper-persistent-data lvm2 #安装必要的一些系统工具**

**2、添加docker yum源并安装docker**

**[root@xuegod63 ~]# systemctl stop libvirtd.service && systemctl disable libvirtd.service && systemctl status libvirtd.service**

**如果有的话，卸载旧的Docker，否则可能会不兼容：**

**[root@xuegod63 ~]# yum remove docker docker-io docker-selinux python-docker-py**

**添加docker-ce yum源配置文件并安装docker**

**[root@xuegod63 ~]# yum-config-manager --add-repo http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo**

**[root@xuegod63 ~]# cat /etc/yum.repos.d/docker-ce.repo #查看新生成的yum源配置文件**

**安装 Docker-CE社区版本**

**[root@xuegod63 ~]# yum -y install docker-ce #咱们在线安装的是docker-ce-18.06.0.ce 版本**

**[root@xuegod63 ~]# systemctl start docker && systemctl enable docker && systemctl status docker #启动Docker服务**

**3、设置docker volume卷挂载方式**

**[root@xuegod63 ~]# mkdir /etc/systemd/system/docker.service.d**

**[root@xuegod63 ~]# tee /etc/systemd/system/docker.service.d/kolla.conf << 'EOF'**

**[Service]**

**MountFlags=shared**

**EOF**

**注：加上MountFlags=shared后，当docker宿主机新增分区时，docker服务不用重启。如果不加docker服务服务重启，docker中的实例才可以使用新加的磁盘或分区。 添加这个参考后，后期在openstack中使用cinder存储服务时，新加磁盘比较方便。**

**4、指定docker 镜像加速器 （很重要，不然后期从国外下载docker镜像会直接报错，而且速度慢）**

**[root@xuegod63 ~]# mkdir /etc/docker/**

**[root@xuegod63 ~]# vim /etc/docker/daemon.json**

**{**

**"registry-mirrors": ["https://e9yneuy4.mirror.aliyuncs.com"]**

**}**

**注：如果需要使用自己的本地私有仓库，写成如下：**

**{**

**"registry-mirrors": ["https://e9yneuy4.mirror.aliyuncs.com"]**

**"insecure-registries": ["192.168.1.63:4000"]**

**}**

**重启相关服务**

**[root@xuegod63 ~]# systemctl daemon-reload #修改了启动脚本，需要执行**

**[root@xuegod63 ~]# systemctl enable docker && systemctl restart docker && systemctl status docker**

**4.6 安装kolla-ansible**

**4.6.1 安装 ansible**

**[root@xuegod63 ~]# yum install ansible -y**

**4.6.1 安装kolla-ansible**

**1、下载kolla-ansible的代码**

**注：kolla-ansible可以实现使用ansible自动安装openstack**

**[root@xuegod63 ~]# cd /root**

**[root@xuegod63 ~]# git clone http://git.trystack.cn/openstack/kolla-ansible -b stable/pike #下载pike版本的openstack**

**2、安装kolla-ansible需要依赖包**

**[root@xuegod63 ~]# cd /root/kolla-ansible**

**[root@xuegod63 kolla-ansible]# pip install . #这里.代表当前目录**

**3、复制kolla-ansible的相关配置文件**

**[root@xuegod63 kolla-ansible]# cp -r etc/kolla /etc/kolla/**

**[root@xuegod63 kolla-ansible]# cp ansible/inventory/\* /etc/kolla/**

**[root@xuegod63 kolla-ansible]# ls /etc/kolla/**

**all-in-one globals.yml multinode passwords.yml**

**注：all-in-one #安装单节点openstack的ansible自动安装配置文件**

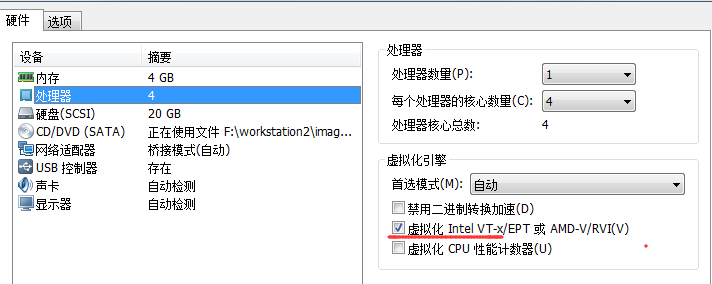
**multinode #安装多节点openstack的ansible自动安装配置文件**

**globals.yml #openstack 部署的自定义配置文件**

**passwords.yml #openstack中各个服务的密码**

**4、修改虚拟机类型为qemu**

**注：如果是在虚拟机里装kolla，希望可以启动再启动虚拟机，那么你需要把virt\_type=qemu，默认是kvm。如果vmware开了“虚拟化Intel VT”功能，就不用写这个了。**



**[root@xuegod63 kolla-ansible]# mkdir -p /etc/kolla/config/nova**

**[root@xuegod63 kolla-ansible]# cat << EOF > /etc/kolla/config/nova/nova-compute.conf**

**[libvirt]**

**virt\_type=qemu**

**cpu\_mode = none**

**EOF**

**4.7 自定义kolla-ansible安装openstack的相关配置文件**

**1、自动生成openstack各服务的密码文件**

**[root@xuegod63 kolla-ansible]# kolla-genpwd**

**[root@xuegod63 ~]# vim /etc/kolla/passwords.yml**

**改：158 keystone\_admin\_password: HsPbEQHxTqmewKYNoRPpIOyQNdEYpHy36OX67TG3**

**为：keystone\_admin\_password: 123456**

**注：这是登录Dashboard，admin使用的密码，你可以根据自己需要进行修改。**

**2、编辑 /etc/kolla/globals.yml 自定义openstack中部署事项**

**[root@xuegod63 ~]# vim /etc/kolla/globals.yml #配置openstack安装中的参数**

**改：15 #kolla\_base\_distro: "centos" #选择下载的镜像为基于centos版本的镜像**

**为： kolla\_base\_distro: "centos"**

**改：18 #kolla\_install\_type: "binary" #去了前面的#号，使用yum安装二进制包安装，源码安装，指的是使用git clone源码安装**

**为：18 kolla\_install\_type: "binary"**

**改：21 #openstack\_release: ""**

**为：openstack\_release: "pike" #指定安装pike版本的openstack，后期下载的openstack相关的docker镜像的tag标记也都为pike**

**23 # Location of configuration overrides**

**24 #node\_custom\_config: "/etc/kolla/config" #配置文件的位置**

**改：31 kolla\_internal\_vip\_address: "10.10.10.254"**

**为：31 kolla\_internal\_vip\_address: "192.168.1.63" # 我们没有启用高可用，所以这里的IP可以和ens33一样，也可以独立写一个和ens33同网段的IP。**

**注：如果配置了高可用，这里要使用一个没被占用的IP。这个IP是搭建HA高可用的浮动IP。 此IP将由keepalived管理以提供高可用性，应设置为和network\_interface ens33 同一个网段的地址。**

**改：73 #network\_interface: "eth0"**

**为： network\_interface: "ens33" # Kolla-Ansible需要设置一些网络选项。 我们需要设置OpenStack使用的网络接口。设置的第一个接口是“network\_interface”。 这是openstack内部多个管理类型网络的默认接口。**

**改：101 #neutron\_external\_interface: "eth1"**

**为：101 neutron\_external\_interface: "ens38" #所需的第二个接口专用于Neutron外部（或公共）网络，可以是vlan或flat，取决于网络的创建方式。 此接口应在没有IP地址的情况下处于活动 如果不是，openstack云平台中的云主机实例将无法访问外部网络。 只要网卡启动着，就可以了，不要给IP，有IP时br-ex桥接就不成功了。**

**135 #enable\_cinder: "no" #先不开启cinder**

**改：151 #enable\_haproxy: "yes"**

**为：151 enable\_haproxy: "no" #去了前面的#号，改yes为no。关闭高可用**

**4.8 开始基于kolla-ansible安装openstack私有云**

**4.8.5 生成SSH Key，并授信本节点：**

**[root@xuegod63 kolla]# ssh-keygen**

**[root@xuegod63 kolla]# ssh-copy-id -i ~/.ssh/id\_rsa.pub root@xuegod63**

**4.8.6 配置单节点清单文件（目前只有一个节点）**

**[root@xuegod63 kolla]# vim /etc/kolla/all-in-one #把localhost替换成xuegod63**

**:1,$s/localhost/xuegod63/**

**:1,$s/ansible\_connection=local//**

**注：将以下组件，都安装到xuegod63这台机器上：改第1行到19行中。**

**[control]**

**xuegod63**

**[network]**

**xuegod63**

**[compute]**

**xuegod63**

**[storage]**

**xuegod63**

**。。。**

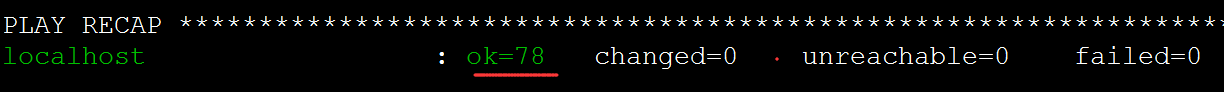
**[deployment]**

**xuegod63**

**4.8.7 开始部署OpenStack**

**1、对主机进行预部署检查：**

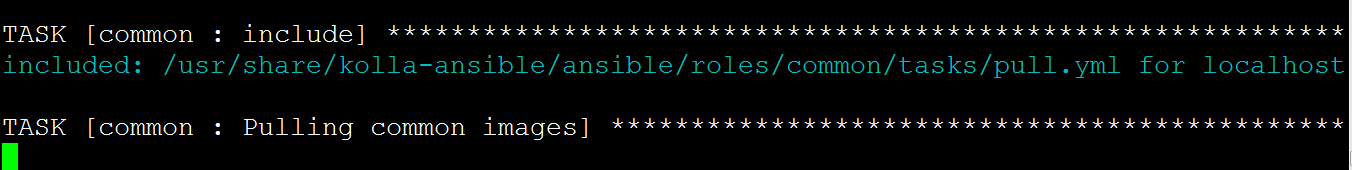
**[root@xuegod63 ~]# kolla-ansible -i /etc/kolla/all-in-one prechecks**

****

**注：我这里有最新显示78个ok，另外，现在还没有运行docker实例了，**

**2、拉取镜像**

**[root@xuegod63 kolla]# kolla-ansible -i /etc/kolla/all-in-one pull**

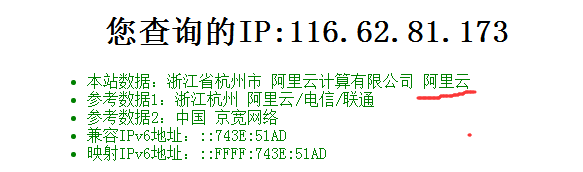
****

**查看网络链接：**

**[root@xuegod63 ~]# netstat -antup | grep 443**

**tcp 0 385 192.168.1.63:34580 116.62.81.173:443 ESTABLISHED 42547/dockerd**

**发现在走的是阿里云的docker加速结点，下载的镜像。**

****

**[root@xuegod63 ~]# docker images #查看下载到的镜像**

**REPOSITORY TAG IMAGE ID CREATED SIZE**

**kolla/centos-binary-cron pike 659fa47c7d43 About an hour ago 455MB**

**。。。**

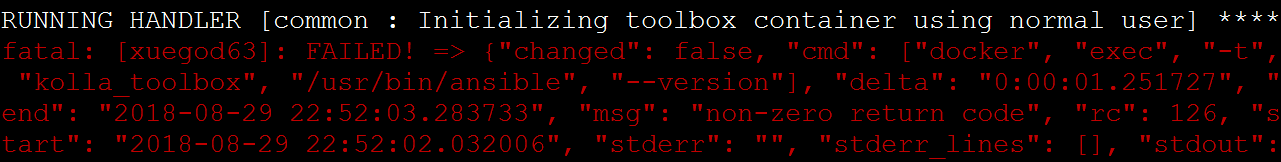
**[root@xuegod63 ~]# docker images | wc -l #整个过程，会下载了32个镜像**

**32**

**3、最后进入实际的OpenStack部署：**

**[root@xuegod63 ~]# kolla-ansible -i /etc/kolla/all-in-one deploy #因为咱们前面已经下载的镜像，所以这时，安装会快一些。如果前面没有下载镜像，那么这时，还会边下载各种openstack相关的镜像边部署docker实例。**

**报错：**

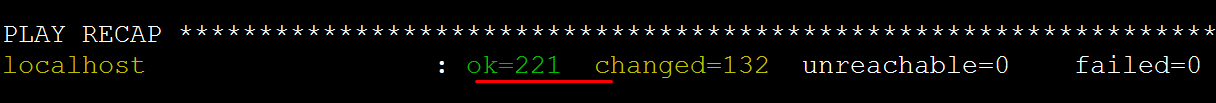


**RUNNING HANDLER [common : Initializing toolbox container using normal user] \*\*\*\***

**fatal: [xuegod63]: FAILED! => {"changed": false, "cmd": ["docker", "exec", "-t", "kolla\_toolbox", "/usr/bin/ansible", "--version"], "delta": "0:00:01.251727", "end": "2018-08-29 22:52:03.283733", "msg": "non-zero return code", "rc": 126, "start": "2018-08-29 22:52:02.032006", "stderr": "", "stderr\_lines": [], "stdout": "OCI runtime exec failed: exec failed: container\_linux.go:348:**

**解决：**

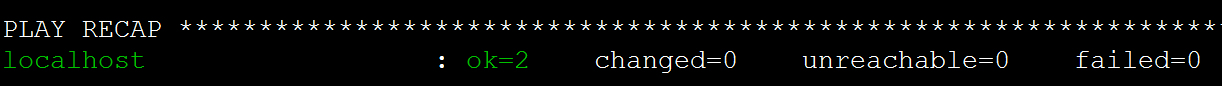
**[root@xuegod63 ~]# kolla-ansible -i /etc/kolla/all-in-one deploy #再运行一下。就自行解决了**

****

**到此安装成功。**

**4、验证部署**

**[root@xuegod63 ~]# kolla-ansible -i /etc/kolla/all-in-one post-deploy**

****

**这样就创建 /etc/kolla/admin-openrc.sh 文件**

**[root@xuegod63 kolla]# ll /etc/kolla/admin-openrc.sh**

**-rw-r--r-- 1 root root 323 8月 10 17:58 /etc/kolla/admin-openrc.sh**

**[root@xuegod63 kolla]# cat admin-openrc.sh #查看openstack登录帐号**

**export OS\_PROJECT\_DOMAIN\_NAME=Default**

**export OS\_USER\_DOMAIN\_NAME=Default**

**export OS\_PROJECT\_NAME=admin**

**export OS\_TENANT\_NAME=admin**

**export OS\_USERNAME=admin**

**export OS\_PASSWORD=123456 #帐号和密码**

**5、测试：http://192.168.1.63 访问成功。**

****

**总结：**

**4.1 openstack 概述**

**4.2 openstack的作用**

**4.3 openstack版本命名规则**

**4.4 openstack各组件关系**

**4.5 Kolla概述和openstack所有结点linux系统初始配置**

**4.6 安装kolla-ansible**

**4.7 自定义kolla-ansible安装openstack的相关配置文件**

**4.8 开始基于kolla-ansible安装openstack私有云**